DLR-IB-RM-OP-2019-194

Recognition and Segmentation of Surgical Gestures

Master's Thesis

Pooja Krishnan

Deutsches Zentrum
DLR für Luft- und Raumfahrt

Institut für Robotik und Mechatronik	BJ.: 2019
	IB.Nr.: **DLR-IB-RM-OP-2019-194**

MASTERARBEIT

SEGMENTATION AND RECOGNITION OF SURGICAL GESTURES

Freigabe:	Der Bearbeiter:	Unterschriften
	Name eintragen	Pooja Krishnan
	Betreuer:	
	Name eintragen	
	Der Institutsdirektor	
	Prof. Alin Albu-Schäffer	Albu-Schäffer

Dieser Bericht enthält 38 Seiten, 19 Abbildungen und 12 Tabellen

Ort: Oberpfaffenhofen	Datum: 11.12.2019	Bearbeiter: Pooja Krishnan	Zeichen: MS

DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Informatics

Recognition and Segmentation of Surgical Gestures Using Recurrent Neural Network

Methode zur Segmentierung von chirurgischen Gesten mit Hilfe von rekursiven neuronalen Netzen

Author:	Pooja Krishnan
Supervisor:	PD Dr. habil. Rudolph Triebel
Advisor:	Antonin Raffin
Submission Date:	13/12/2019

I confirm that this master's thesis in informatics is my own work and I have documented all sources and material used.

Munich, 13/12/2019 Pooja Krishnan

Abstract

Temporal segmentation and recognition of actions performed throughout a video have numerous applications in robotics, medical science, surveillance, etc. It plays a crucial role in the field of Minimally Invasive Robotic Surgery (MIRS), wherein the results can help obscure skill deficiencies, predict the most probable future gesture and improve the quality of feedback provided during surgical training. The current state-of-the-art techniques for MIRS are developed based on kinematic data. However, recent works have found video data to be equally discriminative. In my work, the video-based action segmentation is performed using the Bidirectional Long short-term memory network designed originally for only kinematic data. The model was further improved to make predictions based on both kinematic and video data. Our model achieves competitive performance using both the video and kinematic data on the JHU-ISI Gesture and Skill Assessment Working Set (JIGSAWS). Further, the user is provided with information about the top 3 possible gesture predictions along with an estimate of the model's confidence for each prediction. Additionally, the model was evaluated on a new surgical activity dataset called MIRO dataset, collected using the DLR's MiroSurge System.

Contents

1 Introduction

This chapter discusses the motivation behind choosing this work , the contributions of this thesis and finally, the structure of the following contents is described.

1.1 Motivation

Robotic surgery has exponentially increased over the last decade. The estimated yearly number of procedures has skyrocketed from about 136,000 in 2008 to 877,000 in 2017. The growing demand for robotic surgery or robot-assisted surgery is because it makes minimally invasive surgery possible. Fewer complications, such as surgical site infection, less pain and blood loss, faster recovery and smaller, less noticeable scars are few advantages of this surgical technique. But the technique requires high precision and there are risks involved.

In recent years, recognition of surgical gestures has become an important research area. It helps build systems that assist surgeons depending on the surgical context. For example, it helps improve the quality of surgery performed on critical parts. This is possible due to minimal invasion required to perform the surgery, the precise application of pre-computed pressure on tissues and cuts and the ability of the system to adapt to different situations accordingly. Such surgical procedures can be performed in a minimal amount of time. It reduces the cost of operating rooms, staff and, additionally reducing the amount of anesthetic administered to the patient.However, the task is a safety-critical application. The high accuracy and the estimate of the model's confidence are important. The uncertainty estimation can help capture scenarios where learning is affected by noise or scenarios wherein the model makes predictions on new data with very different patterns from the training data. Estimating the model's confidence can help alleviate risks to a certain extent.The car crash in May 2016 [14] caused because the white side of the trailer and sky could not be differentiated is an example of fatalities that can be caused due to overconfidence.

To improve precision and to overcome risks involved, the surgeons need to be trained and assisted accordingly. Herein lies the benefit of my work. The fundamental objective of the deep learning model is to predict the gestures at each time step accurately. This, in turn, can be used for predicting future gestures while assisting the surgeon,automated surgical skill assessment, and improving efficiency and quality of surgical training. At each time step, the video data and kinematic data help segment and classify the most common gestures used for surgical training such as suturing, knot tying, needle passing, band twist, and weaving.

1.2 Previous Work

Action recognition has many utilities ranging from collaborative robotics to modeling daily life activities. While the classification of some of these tasks like playing basketball etc., can be achieved by capturing contextual information, this technique is not sufficient for fine-grained action recognition. The subtle changes in the object state or location cannot be captured by modeling contextual cues like background appearances etc. The current state-of-the-art model for fine-grained action recognition trained using a video data is a variation of Temporal Convolutional Neural Network called Temporal Deformable Residual Network. Previous, state-of-the-art models for video data were based on Recurrent Neural Networks, Conditional Random Fields (CRF), Bag of Spatio-Temporal Features (BoF) and variations of Temporal Convolution Network. Most of these models decouple the low-level feature representation from the high-level temporal models. For example, in CRF, prediction at each time step is a function of the prediction in the previous time step. In order to learn the nuances of complex actions performed during a surgery, we use a Bidirectional LSTM(BiLSTM). A BiLSTM is capable of capturing latent temporal patterns like the gesture 'reaching for a needle with right hand' will always be performed before the gesture 'positioning the needle'.This is beneficial for both online and offline setup. We also estimate the confidence of each prediction.None of the related works have discussed this previously.

1.3 Contribution

The main goal of this work is to segment and recognize surgical gestures performed in each time step while providing the user with an estimate of the model's certainty. The model is evaluated for the three surgical tasks from JIGSAWS dataset, namely Suturing, Needle Passing and Knot Tying and two surgical tasks from MIRO dataset namely Band Twist and Weaving. The model was trained separately using only kinematic data, only video data and both kinematic and video data to compare contribution of each modality. Additionally, the user is provided with the top 3 most probable predictions for a particular time step. These predictions can be used to improve future predictions and provides the user with an interactive interface with multiple options. Further, the visualizations of uncertainty provide a clear estimate of the model's confidence at the end of each segment and for gestures that are rare and unseen during training.

1.4 Structure

The Introduction (Chap. 1) discusses motivation, contribution, and content structure. Chapter 2, Theory covers the theoretical concepts used throughout this paper. In Datasets (Chap. 3), the datasets used to evaluate the model-JIGSAWS and MIRO are presented in detail. In Chapter 4(Methods), the different network architectures used for segmentation and classification, and the evaluation metrics used are presented in

detail. It also contains the implementation details of the methods discussed, including references to used libraries and pseudo-code.Chapter 5, Results, contains quantitative and qualitative representation of the results for a different combination of data(kinematic and video) and surgical tasks in both the datasets. The chapter also compares the results with the existing state-of-the-art models and discusses the achievements and drawbacks. The final chapter, Conclusion (Chap. 6), summarizes all the chapters and concludes with a brief outlook into future improvements.

2 Theory

This chapter discusses the terminologies and key concepts used in this work. Firstly, we define a Recurrent Neural Network(RNN), its applications and drawbacks. Then the chapter briefs about Long short-term Memory and its variant Bidirectional LSTM. Finally, uncertainty pertaining to deep learning models is discussed.

2.1 Recurrent Neural Networks

Deep learning has numerous applications in the field of computer vision, natural language processing, bioinformatics, gaming, drug design, autonomous driving, etc. Tasks such as image classification, object localization, speech recognition, path prediction, etc., have achieved high accuracy and precision with the advent of neural networks. However, a traditional neural network assumes the data to be independent. It does not draw relations to the events that have occurred in the past. Therefore, a traditional neural network like a feedforward network cannot be used for tasks like the prediction of future frames in case of video processing or words in case of natural language processing.

2.1.1 Definition

Feedforward Neural Networks are computing systems designed to learn specific tasks like classification and regression based on examples. A Recurrent Neural Network is a type of Feedforward Neural Network designed to learn temporal sequences.

"Recurrent neural networks are feedforward neural networks augmented by the inclusion of edges that span adjacent time steps, introducing a notion of time to the mode." – Source: [15]

As depicted in the diagram 2.1, a recurrent neural network contains a memory cell. The output at a current time step h_t depends on the data provided at that time step x_t and output of the previous time step h_{t-1}.This allows the information to flow through the network. A simple recurrent neural network can be represented as follows,

$$A_t = \phi(Uh_{t-1} + Wx_t)$$

Here, ϕ is an activation function like sigmoid or tanh. h_{t-1} and x_t are output from the previous time step and current input data respectively. U and W are weight matrices. These weights are adjusted by backpropagation through time (BPTT) depending on the loss at each time step.If the weights computed at each time step is small then it

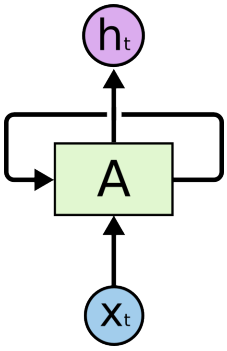

Figure 2.1: Recurrent Neural Network with loop [16]

might lead to a vanishing gradient problem. In deep networks, the gradients shrink exponentially over time as they are backpropagated to the initial layer. This affects the learning of the model as all the gradient to be multiplied are approximately equal to 0. On the other hand, if the values are greater than 1, the values might get larger and eventually lead to an exploding gradient problem. The exploding gradient problem can be handled by clipping off the gradient above a certain threshold.

2.2 Long short-term Memory

A variant of the recurrent neural network is called Long short-term Memory (LSTM)[17]. It was introduced to handle the vanishing gradient problem. An LSTM consists of three gates, namely input, output and forget gate and the cell state. The three gates regulate the flow of information. These gates consist of a sigmoid activation layer and element-wise multiplication. The cell state helps transport information through each unit.

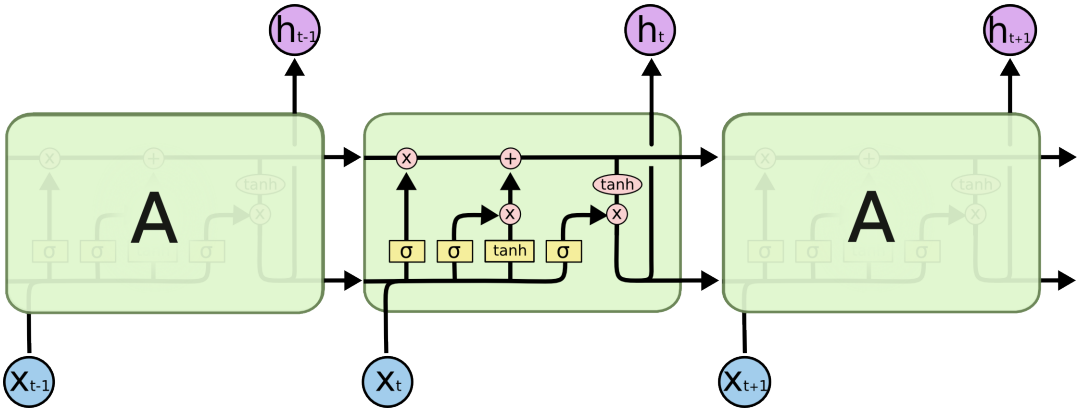

Figure 2.2: Long short-term Memory cell representation [16]

2.2.1 Step-by-step representation

The forget gate is the first step in an LSTM. The decision to erase the previous cell state is made here. This is done with a sigmoid layer. The value '0' indicates that the information should be forgotten and '1' indicates that the information should be kept.

$$f_t = Sigmoid(\theta_{xf} x_t + \theta_{hf} h_{t-1} + b_f)$$

The next step involves deciding what new information should be stored in the cell state. The values that need to be updated are decided by the sigmoid in the 'input gate layer'. This result is multiplied element-wise with the resulting tanh vector.

$$i_t = Sigmoid(\theta_{xi} x_t + \theta_{hi} h_{t-1} + b_i)$$

Based on the amount of previous information we need to store, the cell state is updated.

$$g_t = Tanh(\theta_{xg} x_t + \theta_{hg} h_{t-1} + b_g)$$

In the final step, we filter the output based on the cell state computed. As in the input gate, parts of the cell state that should be retained are filtered using a sigmoid layer. The result of the cell state is passed through a tanh layer and then multiplied with the result of the output gate. This ensures that only the filtered and relevant information is available in the next time step.

$$o_t = Tanh(\theta_{xo} x_t + \theta_{ho} h_{t-1} + b_o)$$

$$Cell : C_t = f_t \otimes Ct - 1 + i_t \otimes g_t$$

$$Output : h_t = o_t \otimes Tanh(C_t)$$

2.3 Bidirectional LSTM

Multiple versions of LSTMs have been designed depending on the application. One such variation is called Bidirectional LSTM. Unlike regular LSTM which is depended only on the past information to make predictions, a Bidirectional LSTM looks at both the past and future. This is done by first training with sequential data as its available and later training on the reversed sequence. This training strategy provides better contextual information and improves learning.

Bidirectional LSTM has multiple applications. An example application is in natural language processing. Translating a sentence from one language to another offline. A Bidirectional LSTM can translate better as it learns the context of the words used both in the past and future.

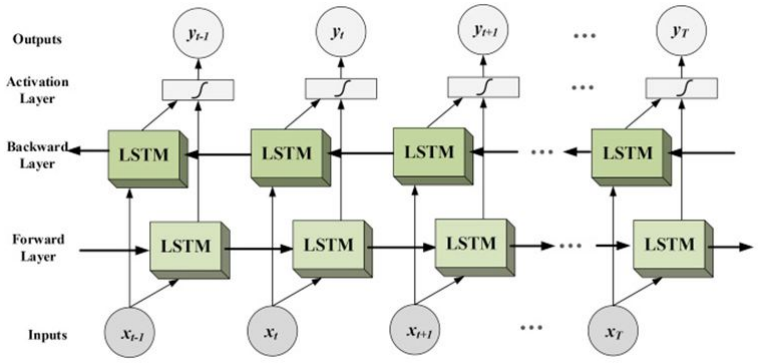

Figure 2.3: Bidirectional LSTM pictorial representation [18]

2.4 Model Uncertainty Estimation

From detecting cancerous lesions to predicting the most viable path for driving an autonomous vehicle – deep learning models have a wide range of applications. However, it's important to understand what the model does not know and scenarios wherein the predictions are not reliable. This is most critical for safety applications like surgical robotics, autonomous driving wherein the repercussions of an uncertain prediction could be disastrous.

2.4.1 Frequentist and Bayesian Statistics

In statistics, there are two different approaches for estimating the parameters and to predict the data. One based on the frequency of the events and others based on the degree of belief. The first approach is followed by the frequentist and the second approach by bayesians. The major difference between Frequentist and Bayesian approach is based on the understanding of probability.The bayesian system can be used to estimate the uncertainty of an event. This approach is useful in case of non-repeatable events.

The Frequentist assume that probability is related to the frequency of occurrence of events. For example, the probability of occurrence of head or tail in the nth trial while tossing a fair coin can be estimated based on the side which appeared most frequently in n-1 trials. The predicted value depends on the distribution of data and can be estimated by computing the maximum likelihood estimate.

However, Bayesian's probability deals with the degree of uncertainty in events. They assume that probability can be estimated based on prior knowledge about the event or data. For example, consider the previous coin scenario, for n = 2; both the trials result in the head then the probability that n=3 trial is also head is 1. This is not possible in the case of a fair coin as the probability for the above scenario ('H' and 'H') is 0.25. Therefore, we need some prior information to calculate the probability.

Bayes rule:

$$P(\theta = x|D) = \frac{P(D|\theta = x)P(\theta = x)}{P(D)}$$

Here, θ is random variable, D is the data, P (θ) is the prior, P(D | θ=x) is the maximum likelihood estimate, P(D) is the evidence and P(θ=x | D) is the posterior.

2.4.2 Aleatoric and Epistemic Uncertainty

We can categorize uncertainty into Aleatoric and Epistemic Uncertainty based on numerous factors.

Aleatoric Uncertainty is also called data uncertainty. It deals with the uncertainty induced by measurement imprecision due to sensor noise, occlusions in image or over-exposure to certain regions in an image, etc. It is an irreducible uncertainty as introducing more data cannot resolve the uncertainty. However, the uncertainty can be reduced to a certain extent by increasing measurement precision. Aleatoric uncertainty can be further classified into:

- Heteroscedastic or Data-Dependent Uncertainty – It is dependent on the input data.
- Homoscedastic uncertainty or Task-Dependent Uncertainty - It remains constant for the entire input data but varies between different tasks. It's useful in case of multitask learning.

Epistemic Uncertainty is also called model uncertainty. It's generally induced by small datasets with sparse training data and safety-critical applications. It is a reducible uncertainty. By adding more data, the epistemic uncertainty can be controlled.

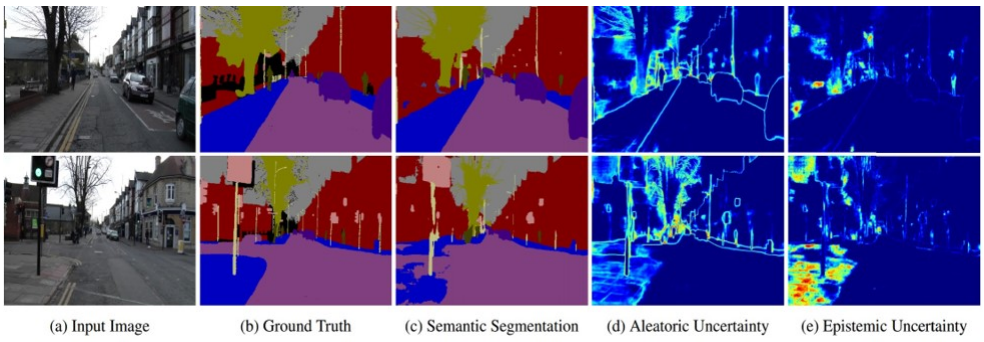

Figure 2.4: Aleatoric vs Epistemic Uncertainty[19]

The image depicts the difference between aleatoric and epistemic uncertainty for semantic segmentation. The aleatoric uncertainty is captured along the object boundaries where labels are noisy. The failure cases like unfamiliar footpath, depicts the epistemic uncertainty.

2.4.3 Estimate classification uncertainty

Unlike regression where the uncertainty can be estimated based on the sample variance of multiple stochastic processes, the classification uncertainty needs alternative statistical measures [20]. In this work, we use predictive entropy [21] to estimate the model's uncertainty. Other known methods are variation ratio and mutual information.

"The predictive entropy captures the average amount of information contained in the predictive distribution." -Source: [20]

$$H[y|x, D_{train}] := -\sum_{c} p(y = c|x, D_{train}) log p(y = c|x, D_{train})$$

If all the classes are predicted with equal probability then the entropy is at its highest value for the data point 'x'. The entropy attains a value of zero i.e. the lowest possible value when one of the classes has a probability of 1. Zero entropy means that the prediction is certain.

2.4.4 Methods to quantify uncertainty

Monte Carlo Dropout

A Bayesian neural network is a neural network with a prior distribution on its weights [22]. It offers robustness to overfitting but with additional computational costs. Therefore, a new method to estimate uncertainty of the model based on the probabilistic interpretation of dropout was introduced.Monte Carlo Dropout (MC Dropout) is based on the principle that Bernoulli approximate variational inference in a Bayesian neural network can be achieved by adding dropout during training and test time[23]. The dropout helps in estimating a probability distribution that is similar to the actual distribution of data. Technically, the dropout minimizes Kullback-Leibler divergence between the posterior of the Gaussian process and the approximate distribution thereby estimating uncertainty in the neural network model. The example below is taken from[23], the images represent the predictive mean and uncertainties on the Mauna Loa CO_2 concentration dataset tested on various models.

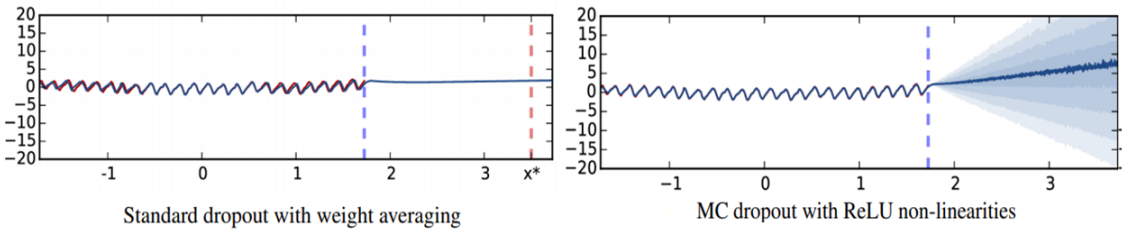

Figure 2.5: MC Dropout on Regression Task [23]

The 'x*' in the image represents a data point far from the actual data. In the standard dropout model, the value 0 is predicted for 'x*' with high confidence even though its

an incorrect prediction. In the second figure, MC dropout captures the uncertainty in prediction (shown in blue; each gradient represents half a standard deviation).

However, there are a few drawbacks to this method. Firstly, the test time is scaled by a factor of the number of forward passes through the network. Second, the dropout distribution does not concentrate with observed data [24]: An agent or model, with dropout enabled for posterior approximation, cannot differentiate between the data point observed once and data points that are repeated. This leads to poor decisions, even when combined with efficient strategies.

Ensemble using bootstrap

In machine learning, the concept of ensembles is used to improve the prediction by using multiple learning algorithms. The variance induced by each of the models in an ensemble can be interpreted as uncertainty. In this work, we created an ensemble using bootstrap sampling. Bootstrap sampling involves random sampling of a dataset with replacement. The technique helps estimate the confidence of the machine learning model for unseen data. The predictions from the multiple models in the ensemble are averaged. The result is then used to compute the predictive entropy. The entropy measure gives a clear perspective on the model's confidence during each prediction.

3 Datasets

3.1 JIGSAWS Dataset

The models are evaluated using the JHU-ISI Gesture and Skill Assessment Working Set (JIGSAWS) [25]. The dataset is the result of a collaboration between John Hopkins University and Intuitive Surgical Inc. The data collected is a temporal sequence of surgical gestures. It includes synchronized video and kinematic data captured using the da Vinci Surgical System. The surgical tasks are performed by eight surgeons possessing varying skill sets.

3.1.1 Surgical Tasks

The elementary surgical gestures performed during the training curriculum are captured in the dataset. According to the paper[25], the surgeons were not allowed to move the camera or apply clutch while performing the surgical tasks. The three elementary tasks are:

- Suturing: The task is performed on a benchtop model with a vertical line (incision) and entry and exit points marked on the artificial tissue. The subject (surgeon) picks the needle and passes it through the entry point and exits it through the exit point marked on the other side of the vertical line. This process is repeated three times.
- Knot Tying: During this task, a flexible tube is attached to a bench-top model. The subject grasps one end of the suture tied to the tube and ties a single loop knot.
- Needle Passing: The subject passes the needle with a suture through four small hoops. The hoops are attached to a small flexible piece above the benchtop model surface. The action is performed from right to left.

3.1.2 Data Description

The tasks in the dataset are performed by eight surgeons with varying robotic surgical skills. Each of the surgeons is indexed as 'B','C','D','E','F','G','H','I'. The surgeons 'D', and 'E' are experienced surgeons. They have more than 100 hrs of robotic surgical experience. Surgeons 'B', 'G', 'H', and 'I' have lesser than 10 hours of experience and surgeons 'C' and 'F' have between 10 and 100 hrs of experience. All of the chosen surgeons are right-handed.

Figure 3.1: Three elementary tasks(left to right):Suturing, Knot-Tying and Needle Passing [25]

Each task was performed five times. The paper [25] refers to each of these tasks as a 'trial'. Few trials recorded were rendered unusable due to data corruption. Therefore, there are only 39 Suturing trials, 36 Knot Tying trials and 28 Needle Passing trials in the JIGSAWS dataset.

The dataset consists of both kinematic and video data for all the above-mentioned trials. Both the kinematic and video data are synchronized and manually annotated. The list of manually annotated gestures in JIGSAWS dataset are given in Tab.3.1. The image below illustrates the frequency of each gesture for the suturing task.

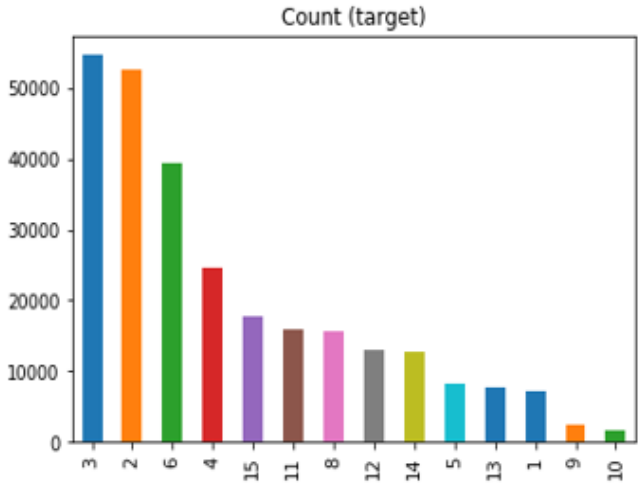

Figure 3.2: Frequency of each gesture performed during Suturing task

- Kinematic data: The data is captured using the API from the da Vinci Surgical System(dVSS) at 30Hz. The dVSS consists of two manipulators(MTMs) at the master side console used by the surgeons and three patient side manipulators (PSMs). The motion of the left and right MTMs and first and second PSMs are described using 19 kinematic variables. Therefore, the kinematic data consists of 76 features.

- Video data: The dVSS consists of two endoscopic cameras mounted on the patient side. The data is captured at a sampling rate of 30Hz and a resolution of 640x480. The video captured by the left and right camera is stored as 'capture1' and 'capture 2' files. The data is encoded using four-character code (FOURCC) and stored in AVI format.

Gesture Index	Gesture Description
G1	Reaching for needle with right hand
G2	Positioning needle
G3	Pushing needle through tissue
G4	Transferring needle from left to right
G5	Moving to center with needle in grip
G6	Pulling suture with left hand
G7	Pulling suture with right hand
G8	Orienting needle
G9	Using right hand to help tighten suture
G10	Loosening more suture
G11	Dropping suture at end and moving to end points
G12	Reaching for needle with left hand
G13	Making C loop around right hand
G14	Reaching for suture with right hand
G15	Pulling suture with both hands

Table 3.1: Gestures in JIGSAWS dataset[25]

3.2 MIROSurge Dataset

In addition to the JIGSAWS dataset, the models were also tested using MIROSurge dataset[26]. The dataset consists of both kinematic and video data logged at a rate of 30Hz. The MiroSurge dataset contains recordings collected from 10 users over a period of 3 weeks. Three main surgical tasks in the dataset are:

- Band Twist: In this task, the subject moves the ring placed on the rods at one side of the box to other side and back.In order to pick or place the ring, the elastic band attached must be pulled towards the center of the box.

- Weaving: The task is performed on a modified Weaving training module from the Lübecker Toolbox. The component consists of four elastic bands placed equidistant from each other. The subject weaves the thread back and forth among the bands.

- Ring Passing: It's a naive task introduced to get the user comfortable with actions like translation, rotation and other such manipulation of instruments and also the

indexing mechanism used on console. The goal of the task is to transfer the two semi-transparent rings to the rods on opposite corners and back.

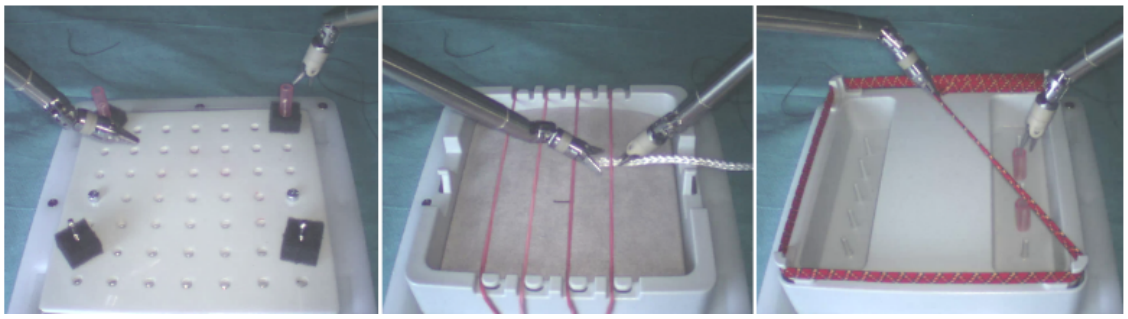

Figure 3.3: Three elementary tasks(left to right):Ring Passing, Weaving and Band Twist [26]

3.2.1 Data Description

The kinematic data contains features collected from both the master and slave side. At the master side, the gripper angles and, the position and orientation of the tool tips are recorded. The velocity, orientation and rotational velocity of the tool tip and the gripper angles are recorded at the slave side.

For every trial, two video recordings are available. One video is recorded by the endoscopic camera and another by webcam.The timestamp from the kinematic data is synchronized with the time in video recordings on a frame level. The synchronization is done using the Network Time Protocol(NTP).The dataset is manually annotated at the frame level with gesture labels for weaving and band twist indicated in Tab 3.2 and Tab 3.3

Gesture Index	Gesture Description
G1	Reaching for the thread
G2	Grasping
G3	Going down the first band
G4	Going over the second band
G5	Going down the third band with the thread
G6	Helping the pull
G7	Reorienting the thread
G8	Picking up the thread and holding until left hand picks it up
G9	Pulling the thread to the right
G10	Going to end point

Table 3.2: Gestures in MiroSurge dataset for Weaving task [26]

Gesture Index	Gesture Description
G1	Going to pin 1
G2	Grasping
G3	Pulling up pin 1
G4	Moving to slot 1
G5	Going to pin 2
G6	Going to slot 2
G7	Going to left band
G8	Pulling left band upwards
G9	Releasing left band from up
G10	Going to right band
G11	Pulling right band upwards
G12	Releasing right band from up
G13	Pulling left band downwards
G14	Releasing left band from down
G15	Releasing right band downwards
G16	Releasing right band from down
G17	Going to end point

Table 3.3: Gestures in MiroSurge dataset for Band Twist task [26]

4 Methods

In this chapter, we discuss the methods using to segment and recognize surgical gestures.The data augmentation technique applied and the results of different feature extraction techniques are discussed in detail. The chapter then discusses the various neural network architectures experimented, their implementation details and the various hyperparameters used. Finally, we discuss the algorithms used to estimate the network's confidence.

4.1 Data Augmentation

The success of deep learning in computer vision tasks like image classification can be attributed to the availability of large amounts of data. However, the process of collecting and labeling huge datasets is expensive and not always feasible. A more cost-effective approach to increasing the diversity of data is called data augmentation. For tasks like classification, augmentation can be achieved by performing geometric transformations like cropping, translation, rotation, resizing, flipping, etc.

In this work, data augmentation was performed to curb the overfitting problem. Both JIGSAWS and MiroSurge datasets contain a limited amount of data with certain gestures performed more frequently than others. Since the dataset is small, we perform offline data augmentation. During offline augmentation, each transformation performed is stored on the disk. We perform transformations like scaling, translation, rotation in sequence to create the first set of augmented data. The second set of augmented data was generated by performing cropping, flip, and Gaussian blur. The transformations were chosen such that they generate scenarios wherein the camera is placed in a wrong position or the lighting in the surgical room is not apt.

4.2 Feature Extraction

Feature extraction is a technique applied to achieve dimensionality reduction. The process extracts relevant information from the data thereby, reducing the amount of data that needs to be processed and removes redundant information. Identifying relevant information is important as it reduces the number of variables to be learned and the need for more computing power. It also helps prevent overfitting on training samples and helps build more generalized models.

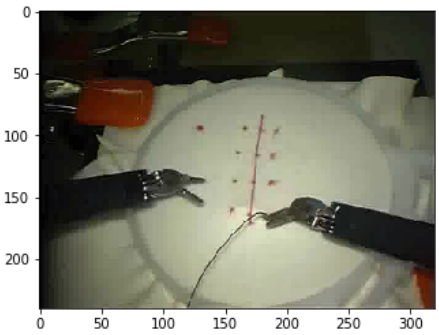

Figure 4.1: Original Image

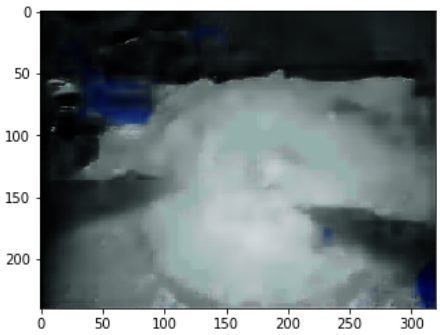

Figure 4.2: Decoder output image

Several known techniques perform feature extraction. In this work, we have conducted experiments using Convolutional Autoencoder, Scale Invariant Feature Transform (SIFT) and Inception V3 pre-trained on Imagenet to extract features. The convolutional autoencoder is a dimensionality reduction technique which encodes the data by training the network to ignore the noise signals. The decoder then tries to regenerate the original image from the encoded representation. The regenerated image 4.2 is an approximate representation of the input as only the relevant aspects of the original data are retained. The second method, SIFT was employed in most of the previous works on gesture recognition. The SIFT technique finds keypoint in the images 4.3. Each keypoint is a special structure with attributes like (x,y) coordinates, size of the neighborhood, orientation angle, strength of the keypoint, etc,[27]. The features extracted using Convolutional Autoencoder and SIFT are fed into the Bidirectional LSTMBidirectional LSTM. The resulting accuracy in both cases was about 30% .

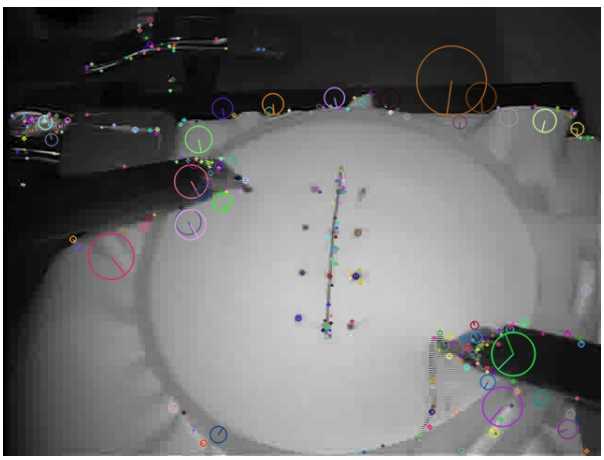

Figure 4.3: SIFT keypoints

We use Inception_V3 model pretrained on Imagenet database to extract features. The image of size 640×480 was resized to 299×299 based on nearest pixels. The

augmented images and the real image are passed through the model. The resulting extracted features are stored in the disk.

4.3 Neural Network Models

We experimented with numerous models like Conv3D, variations of LSTM. Below we discuss the most important network architectures that provided optimum results for various combinations of kinematic and video data.

4.3.1 BidirectionalLSTM

The extracted kinematic/video features are normalized and then fed as input into forward LSTM first. The re-verse order of the input is fed into reverse LSTM. The two results are then concatenated. The logits are computed based on the concatenated results. The resulting logits are provided as input to softmax function to retrieve probabilities for each gesture. The network consists of one LSTM block and uses an initial learning rate of 2.0. Since the data is sequential, we use the concept of sweep instead of epochs. A sweep is a collection of batch size sequences that continue until all sequences in the batch are exhausted. Short sequences are handled by wrapping around time.For example, consider the following input sequence [[0,1],[1,0,1]]. Considering the batch size for processing the sequence is set to 3. The first sub-sequence is short and the batch size is 3, so we need to generate a new sequence by wrapping around time. The resulting new sequence is [[0,1,0],[1,0,1],[0,1,0]]. The network is trained for 1200 sweeps. The batch size is set to 3. Gradients are computed using stochastic gradient descent and the loss is calculated using softmax cross entropy and exponential moving average.The exponential moving average (EMA) is applied to compute the weighted moving average that gives more importance to recent changes in gestures.To prevent overfitting a L2 regularization is added to the loss. The hyperparameter are optimized through trial-and-error method.The source code available in [28] was adapted for all experiments.

4.3.2 Multi-Layer Perceptron + Bidirectional LSTM

To train both kinematic and video data the network architecture discussed in the previous section is slightly modified. A Multi-layer perceptron consisting of two hidden layers is added to process the kinematic data. The processed kinematic data is concatenated with the video data and passed as input to the Bidirectional LSTM. All 76 kinematic features in case of JIGSAWS and 49 features in case of MiroSurge dataset were used as input to the multi-layer perceptron.

4.3.3 Other Related Networks

In this subsection, we review the other action segmentation-based network architectures we tried to reproduce the results for. Both the architectures discussed are variations of Temporal Convolutional Network(TCN).

Encoder-Decoder Temporal Convolutional Network(ED-TCN)

As depicted in the figure 4.4, the encoder-decoder network uses a sequence of temporal convolutions and temporal pooling/unpooling layers to process a stream of input. According to the paper [29], the input to the TCN is a set of video features, such as output from temporal or spatiotemporal CNN for each frame of given video. We experimented with the publicly available source code for ED-TCN network. Input to the network was features extracted from InceptionV3 pretrained on Imagenet. The accuracy recorded in the paper is 81.4%.

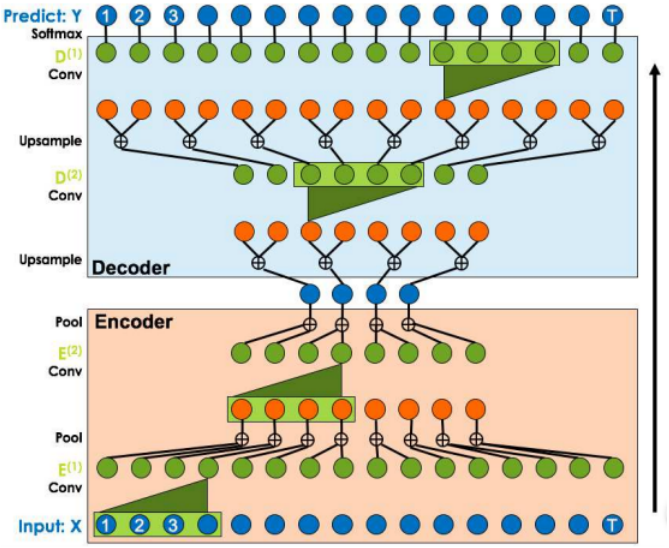

Figure 4.4: ED-TCN Network Architecture [29]

Temporal Deformable Residual Network(TDRN)

The paper [30] addresses the task of action segmentation by accepting CNN features for each frame as input. TDRN consists of two processing streams: Residual stream that analyzes video information for accurate action segmentation and pooling/unpooling stream that captures temporal aspects in each frame for precise action recognition. The two streams are fused together through a sequence of Deformable Temporal Residual Module(DTRM) as shown in figure. The authors have recorded an accuracy of 84.6% in the paper on performing 8-fold-cross validation with 39 suturing videos. However, due

to lack of clarity in the architecture of the network (DTRM module- contacted authors but no response) we did not achieve required accuracy.

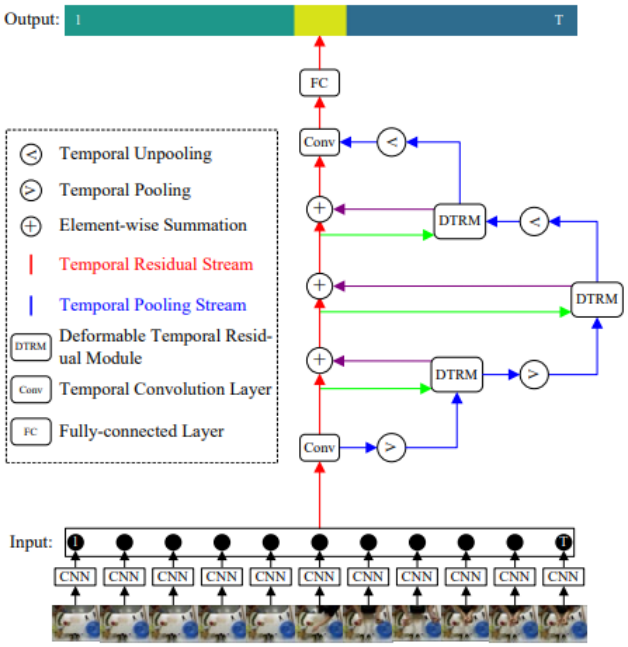

Figure 4.5: TDRN Architecture [30]

4.4 Evaluation Methodology

To evaluate the algorithms, the JIGSAWS paper[25] suggests cross validation setting discussed below:

- Leave-one-super-trial-out (LOSO): A super trial is the i-th trial performed by all the subjects. The JIGSAWS dataset contains 5 trials for almost all users. To perform LOSO setup for cross validation, 5 folds are created with each fold consisting of the i-th trial performed by the subject as a test set. In case of the MiroSurge dataset, 3-fold cross valdiation was performed. The LOSO experimental setup helps validate the robustness of the algorithm when i-th trial is skipped for every subject.
- Leave-one-user-out (LOUO): The JIGSAWS dataset contains of 8 users. To perform LOUO setup for cross validation, 8 folds are created with each fold consisting of all the trials performed by the i-th user as a test set. For MiroSurge dataset, the folds are chosen depending on the availability of data. The LOUO experimental setup helps validate the efficiency of the algorithm to detect the gestures performed by a new or unseen subject.

The performance measures of both LOSO and LOUO is calculated using accuracy and edit distance.

- Accuracy: The fraction of samples that the model predicted correctly.

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{total number of predictions}}$$

- Edit Distance: In this work, we use the Levenshtein distance to measure the differences between two sequences. It's achieved by quantifying the minimum number of insertions, deletions, and substitutions required to transform one sequence of gestures to others.

Additionally, our network also predicts the top three possible gestures for a frame. The top three predictions are included to give the surgeon a choice to choose from the set of most probable gestures. These suggestions are useful for surgeons when online surgery is conducted. Based on the previous surgical gestures performed the surgeons can choose the next most possible gesture. The accuracy for each level of top-n prediction is estimated.

4.4.1 Model Uncertainty Estimation

Many modern applications achieve state-of-the-art performance without accounting for the model's confidence. It is important to understand what a model does not know. For example, in classification tasks, the softmax probabilities are erroneously estimated as the model's confidence. A model can be uncertain about a prediction with very high softmax probability. In the sections below, we discuss the different confidence estimation methods applied to estimate the model's certainty while predicting a gesture.

Monte Carlo Dropout

To estimate the classification confidence of the model, we trained Bidirectional LSTM with dropout applied before every weight layer. The dropout probability is set to 0.5. The trained model is then evaluated using the test set. Dropout is also enabled at during inference. We performed 50 stochastic forward passes. During every stochastic forward pass, for each sample 'x' different dropout masks are applied to retrieve the inference. The average of the 50 predictions is used to measure the predictive entropy. This provides an estimate of the uncertainty for each prediction.

Naive Ensemble

In this approach, we train multiple Bidirectional LSTM models and combine the predictions of all the models. This technique is called ensemble learning. The variance

induced by the ensemble can be viewed as an uncertainty. The following steps were followed to estimate the uncertainty using ensembles[31]

Algorithm 1: Uncertainty Estimation using Naïve Ensemble

Result: Uncertainty estimation
Initialize the parameters $\theta_1, \theta_2, \theta_m$ with random values;
Train each of the $m \epsilon M$ networks with the θ_m values;
Combine the predictions: $p[y|x] := \frac{\sum_{m=1}^{M} p_{\theta_m}(y|x, \theta_m)}{M}$

Bootstrap Ensemble

As discussed in section 2.4.4, an ensemble is created by sampling the data with replacement. To map the results with the cross-validation methodology applied for evaluation, the following approach was used to compute the uncertainty.

Algorithm 2: Uncertainty Estimation using Bootstrap Ensemble

Result: Uncertainty estimation
Select one user of M users as the test user;
Train each of the $m \epsilon M$ models by leaving one user out;
for *m in 1,..M-1* **do**
  $P(y|x) := f_{\theta m}(x)$
end
$P(y = c|x) = \frac{\sum_{m=1}^{M-1} \sum_c P(y|x)}{(M-1)}$;
$H[y|x, D_{train}] := -\sum_c p(y = c|x) log p(y = c|x)$;

The algorithm estimates the uncertainty for each prediction by training M-1 models, where M is the number of users in the dataset. Each model is trained by excluding one user. This ensures that user data is sampled with replacement. To estimate the entropy, the average of the softmax probabilities for all classes is computed and substituted into the entropy equation[2.4.3]. Similarly, M-1 models are trained, where M is the number of trials for each user, to estimate uncertainty when one trial is left out.

Ensemble with Randomized Prior

The drawback of using dropout for posterior approximation is that the rate of the dropout does not depend on the data.This can affect the accuracy of the model even if we use efficient network structures. Therefore, we estimate the model's uncertainty by introducing random prior in an ensemble. According to the paper [32], the final model $Q_{\theta_k}(x)$ is an ensemble of 'k' models is created by adding untrained random prior value $p_k(x)$ to the function trained to fit the data $f_{\theta_k}(x)$.

$$Q_{\theta_k}(x) = f_{\theta_k}(x) + p_k(x)$$

5 Results

The chapter discusses the results collected from the implementation of methods discussed in the previous chapter 4.Results are presented in both quantitative and qualitative format . The tables discuss the comparison between each gesture and the current known state-of-the-art, comparison of results with different combination of kinematic and video data and top-3 possible gesture predictions. The visualization for the suturing task for all the three types of the data are added in this chapter.The visualization includes the confidence estimation using MC Dropout. The results of confidence estimation performed using the naïve ensemble, ensemble with bootstrap and ensemble with random prior for suturing task are also included.

The chapter is divided into two sections to present the results for the two datasets. The first section, discusses the results for JIGSAWS dataset and the second section discusses results based on the MiroSurge dataset. The two results are not directly comparable as the amount the data available for training in MiroSurge Dataset is minimal and the gestures performed are different.

5.1 Results for JIGSAWS dataset

In this section, we discuss the results retrieved for gestures performed in JIGSAWS dataset.

5.1.1 Comparison of Methods

In this section, we first compare the results of Bidirectional LSTM model with the known current state-of-the-art techniques. The results with respect to all possible combination of data is also included. Accuracy and edit distance are used as key performance indicators. The results are presented for both the leave-one-user-out (LOUO) and leave-one-trial-out (LOTO) cross-validation formats.

Comparison with state-of-the-art						
Models	Suturing		Knot Tying		Needle Passing	
	Accuracy	ED	Accuracy	ED	Accuracy	ED
CRF[33]	68.8	NA	60.17	NA	54.52	NA
Semi CRF[33]	59.41	NA	41.46	NA	46.89	NA
MsM CRF[33]	71.76	NA	66.94	NA	60.39	NA
Bidirectional LSTM	76.2	12.14	70.44	23.33	49.95	85.18
ED-TCN[29]	81.4	11.1	NA	NA	NA	NA
TDRN[30]	84.6	10.2	NA	NA	NA	NA

Table 5.1: LOUO trained with video data

Comparison with state-of-the-art						
Models	Suturing		Knot Tying		Needle Passing	
	Accuracy	ED	Accuracy	ED	Accuracy	ED
MsM-CRF(kin-STIP)[33]	70.09	NA	68.43	NA	54.41	NA
MsM-CRF(kin-dense)[33]	72.6	NA	68.83	NA	57.08	NA
MLP+Bidirectional LSTM	81.38	10.51	75.57	20.33	68.55	49.42

Table 5.2: LOUO trained with video and kinematic data

Comparison with state-of-the-art						
Models	Suturing		Knot Tying		Needle Passing	
	Accuracy	ED	Accuracy	ED	Accuracy	ED
MsM-CRF[33]	69.03	NA	64.28	NA	52.39	NA
GMM-HMM[33]	73.95	NA	64.13	NA	72.47	NA
Forward LSTM	80.5	NA	NA	NA	NA	NA
Bidirectional LSTM	82.03	6.24	83.15	9.5	77.21	15.82
MS-RNN	90.2	NA	NA	NA	NA	NA

Table 5.3: LOUO trained with kinematic data

The tables depict the Accuracy and Edit Distance(ED) for segmenting and recognizing gestures performed during suturing tasks using only video data, video and kinematic, and only kinematic data. The model in bold is the model trained as a part of our work. All three types of data are trained to leave one user out.

In the case of the video data, we tried to reproduce the results for both ED-TCN and TDRN but achieved an accuracy of 78% and 76% respectively for suturing task trained by leaving a user out. We assume the difference is results in the case of ED-TCN is due to differences in the method used for feature extraction. In the case of TDRN, due to a lack of clarity (contacted the authors but did not receive any response)in the architecture of the network (DTRM module), our implementation achieved an accuracy of only 76%.

The model trained on kinematic and video data has slightly lower accuracy than the model trained on only kinematic data. The initial assumption was that the model would learn better with more data. To analyze this condition further we trained both kinematic and video data on only Bidirectional LSTM. The data from both kinematic and video was directly concatenated and provided as input. The resultant accuracy was about 81%. Therefore, we need to apply better techniques to extract video features that complement kinematic features. This might help improve the accuracy of the model trained on kinematic and video data.

Comparison with state-of-the-art						
Models	Suturing		Knot Tying		Needle Passing	
	Accuracy	ED	Accuracy	ED	Accuracy	ED
Semi CRF[33]	65.83	NA	44.82	NA	56.22	NA
CRF[33]	76.51	NA	69.16	NA	62.23	NA
MsM CRF[33]	79.04	NA	72.04	NA	68.81	NA
Bidirectional LSTM	81.17	8.078	76.24	19.014	56.73	50.36

Table 5.4: LOTO trained with video data

Comparison with state-of-the-art						
Models	Suturing		Knot Tying		Needle Passing	
	Accuracy	ED	Accuracy	ED	Accuracy	ED
MsM-CRF(kin-STIP)[33]	82.49	NA	80.5	NA	76.41	NA
MsM-CRF(kin-dense)[33]	82.81	NA	81.11	NA	76.82	NA
MLP+Bidirectional LSTM	86	6.126	82.052	10.68	75.95	32.03

Table 5.5: LOTO trained with video and kinematic data

Comparison with state-of-the-art						
Models	Suturing		Knot Tying		Needle Passing	
	Accuracy	ED	Accuracy	ED	Accuracy	ED
MsM-CRF[33]	80.99	NA	79.39	NA	74.85	NA
sparse-HMM	81.1	NA	82.6	NA	76.1	NA
Bidirectional LSTM	87.09	3.8	86.5	6.37	79.61	12.53

Table 5.6: LOTO trained with kinematic data

The tables above depict the results for experiment performed by leaving one trial out. The accuracy for leave-one-trial-out is higher than the leave-one-user-out because some users have longer recorded sequences compared to others, by skipping a trial the amount of data available for training is not affected.Also, some gestures like 'G9' - 'Using right hand to help tighten suture' is only performed by experienced surgeons

such as surgeon 'D', such gestures cannot be learnt by the model when trained by leaving the user 'D' out. This affects the accuracy of the model.

5.1.2 Top-n Accuracy

In this subsection, we discuss the results for top-n accuracy. The results for all the JIGSAWS task and all combinations of data can be found in 5.7.

Top-n accuracy						
Tasks	Top-1		Top-2		Top-3	
	LOUO	LOTO	LOUO	LOTO	LOUO	LOTO
Suturing -vid	76.2	81.176	89.96	92.30	93.435	94.789
Suturing -vid+kin	81.38	86	91.537	93.73	93.61	95.23
Suturing -kin	82.03	87.09	91.86	94.67	94.39	96.38
Knot Tying -vid	70.44	76.24	75.63	89.88	92.16	93.79
Knot Tying -vid+kin	75.57	82.052	89.84	92.9	94.93	96.36
Knot Tying -kin	83.15	86.5	93.57	94.84	96.67	96.84
Needle Passing -vid	49.95	65.12	70.458	80.51	81.18	90.2
Needle Passing -vid+kin	68.5	75.95	84.74	88.3	90.18	91.67
Needle Passing -kin	77.21	79.61	88.96	89.76	92.29	92.44

Table 5.7: Top-n accuracy

5.1.3 Timeline Visualization

This section contains images representing the timeline of gestures performed by user 'D' The first line represents the ground truth, the second line represents the most probable gesture prediction, the third line represents the second most probable, fourth line represents the third most probable gesture and final line represents the uncertainty measured using MC Dropout. Each color in the top 4 rows represents a gesture. In the final row, white color depicts that the model is highly confident and different gradients of red color showcases models uncertainty. Darker the red is, more uncertain the model is about its most probable prediction.

The image 5.1 is timeline visualization for user 'D' as the test set. As discussed earlier, the gesture 'G9' is only performed by experienced surgeons. It is evident in the image that this gesture is not learned by the model. The uncertainty is high around this gesture and in areas where the prediction is wrong. Further, the images in the inset are frames for the 'G6' and 'G9'. The only difference between the two images is the position of the needle and right arm. Such minute difference may not be immediately captured by the model. This might lead to errors along with the start and stop of the segments.

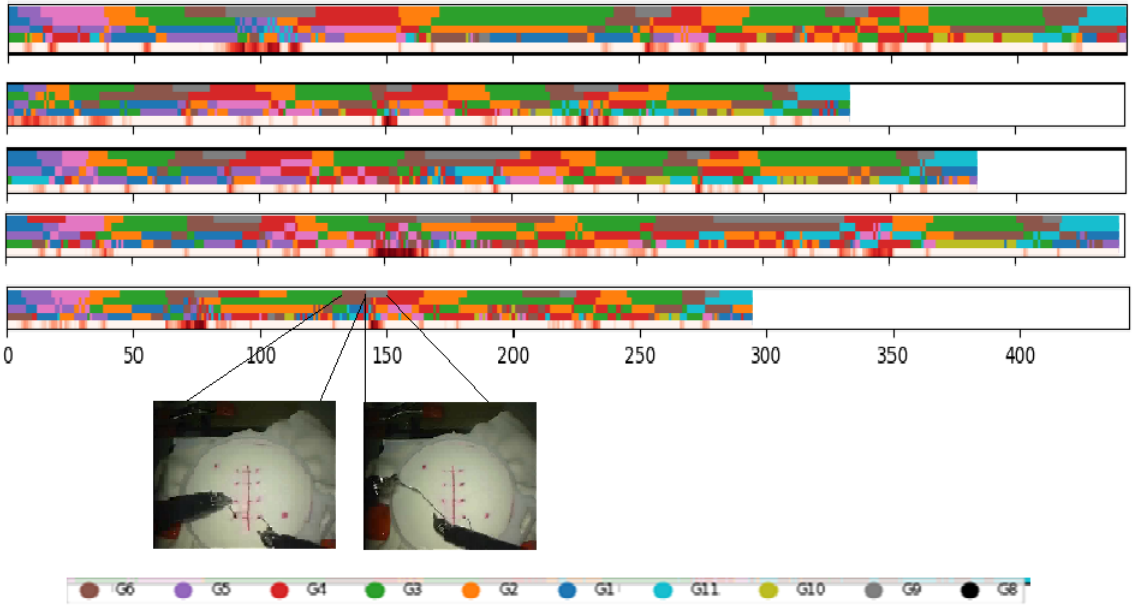

Figure 5.1: Suturing Task - LOUO

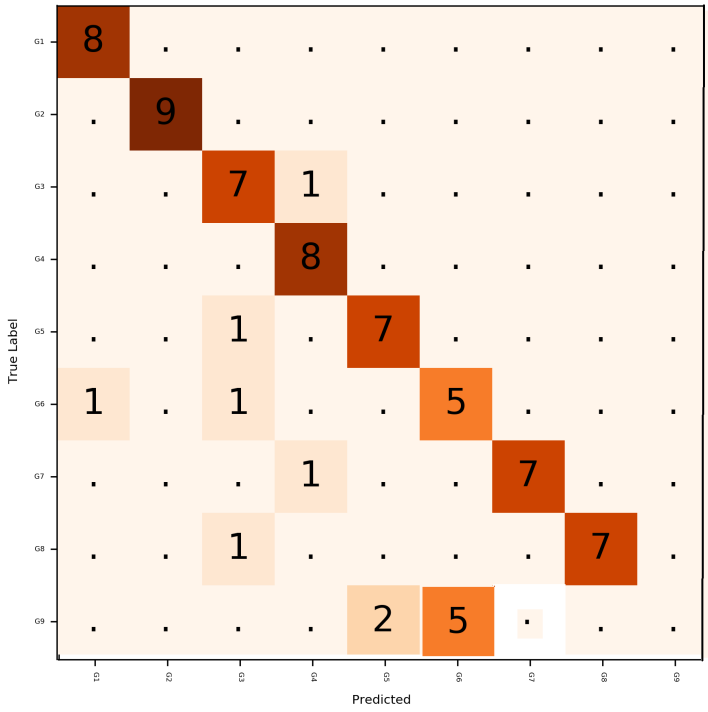

Figure 5.2: Confusion Matrix for trial 5

The figure 5.2 represents the confusion matrix for the above visualization. As discussed above the model fails to learn the 'G9' gesture. Its found to be most similar to 'G5' and 'G6' gestures.

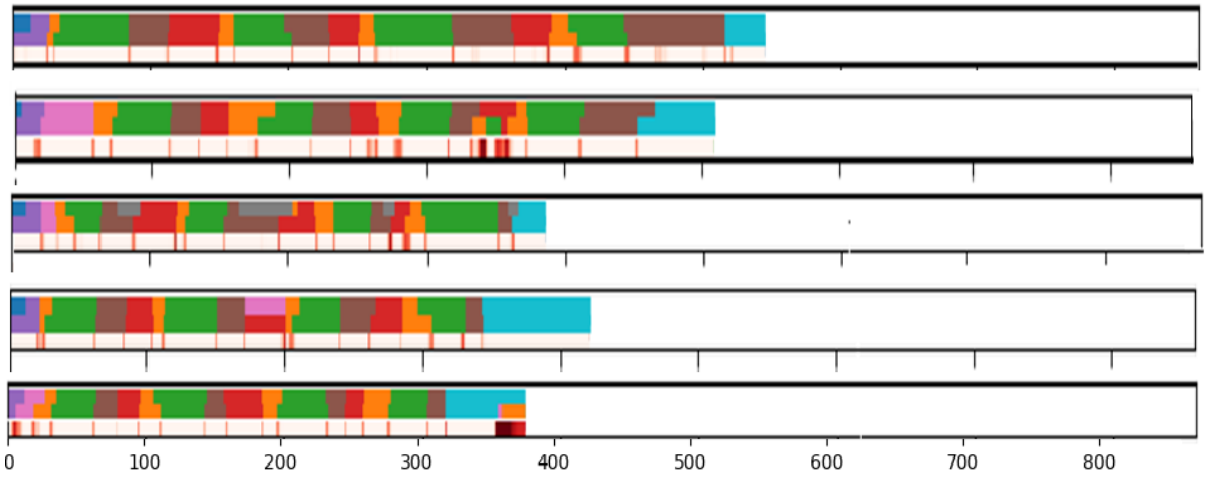

Figure 5.3: Suturing Task - LOTO

The figure 5.3 illustrates the visualization for trial 3 of all users.The uncertainty is high in areas of wrong prediction.

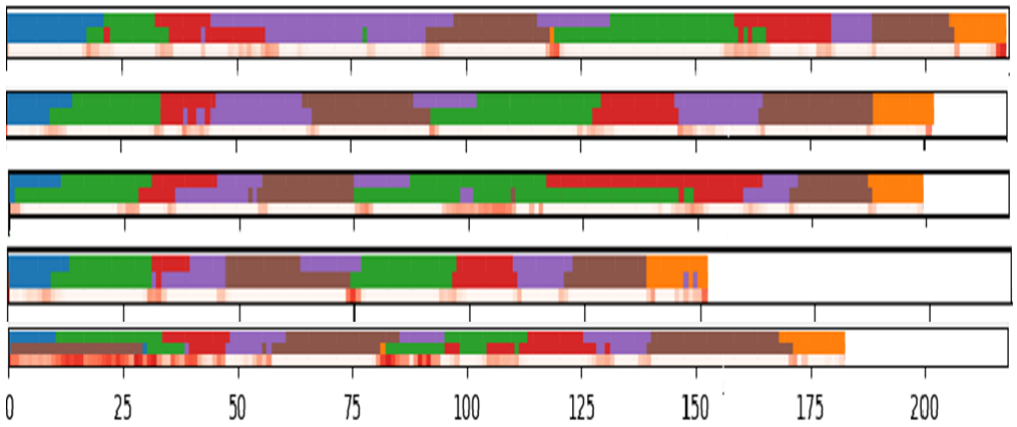

Figure 5.4: Knot Tying Task - LOUO

The figure 5.4 represents the timeline visualization for knot tying task. The experiment is performed by leaving user 'D' out.

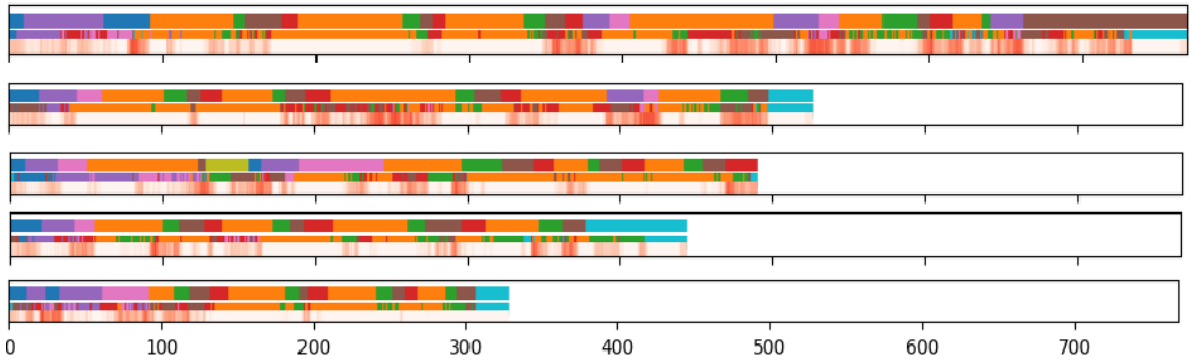

Figure 5.5: Needle Passing - LOUO

In figure 5.5, the orange gesture namely, 'G2'- Positioning Needle is performed more often. The model learns this gesture the best. The amount of data available for training Needle Passing task is less as user 'G' data is not available. This affects the overall accuracy of prediction during this task. The MC Dropout can capture the major deviations in the predictions.

5.1.4 Uncertainty estimation using Ensembles

In this section, we discuss the results of uncertainty estimated using different ensembles.

Naive Ensemble

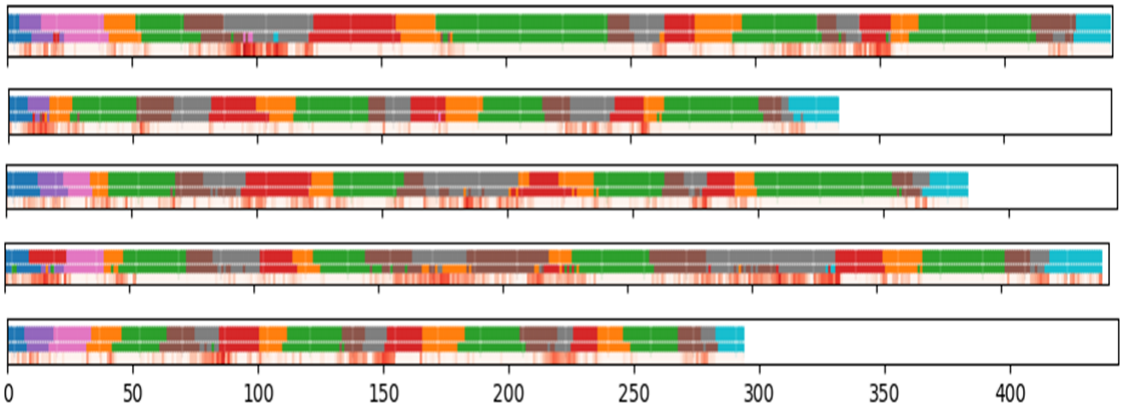

Figure 5.6: Uncertainty estimation using Naive Ensemble

The figure depicts the result for user 'D'.The naive ensemble captures all the epistemic uncertainty. In most of the cases,the uncertainty is high for grey gesture '*G9*'. The model is uncertain about its prediction even though its correct prediction. This might be due to fact that 'G9' is a very rare gesture.

Random Prior

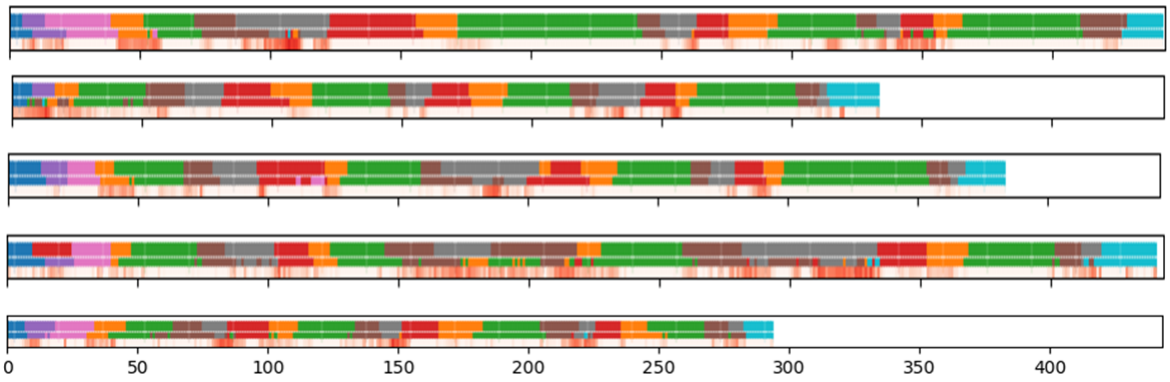

Figure 5.7: Uncertainty estimation using Random Prior

The image depicts the method discussed in section 4.4.1. The random prior detects epistemic uncertainty.But it is unable to capture the aleatoric uncertainty i.e, the uncertainty along the segment edges.

Bootstrap

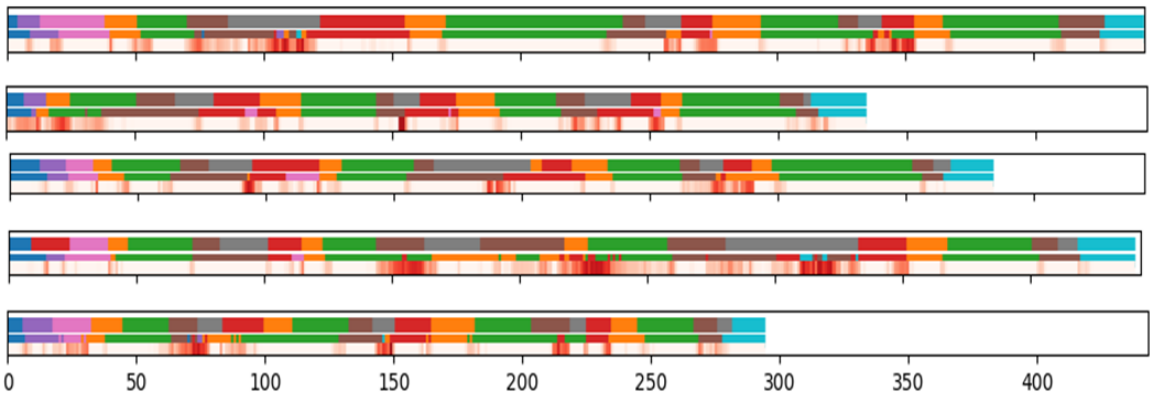

Figure 5.8: Uncertainty estimation using Bootstrap

The Bootstrap model can capture both epistemic and aleatoric uncertainty. The ensemble can estimate the uncertainty along the transitions from one gesture to another. And, like rest of the uncertainty models its also able to capture the error situations.

5.2 MIROSurge Dataset

In this section, we discuss the test results for MIROSurge Dataset for all combinations of data and for both LOUO and LOTO.

5.2.1 Comparison of Methods

Here we compare the results with the known state-of-the-art method. Our model is highlighted in bold.

Comparison of Methods				
Methods	Band Twist		Weaving	
	LOUO	LOTO	LOUO	LOTO
MPL C[26] -kin	17	25	31	42
Bidirectional LSTM - kin	35.15	39.23	62.51	65.2
Bidirectional LSTM - kin+vid	68.052	76.51	73.95	75.12
Bidirectional LSTM - vid	66.52	72.54	71.38	73.2

Table 5.8: Comparison of Methods for MIRO dataset

Unlike the JIGSAWS, the MIROSurge dataset provides better results by using both kinematic and video data. This could be because the kinematic data in MIROSurge dataset has only 49 features. And, the amount of data available for training is also less compared to JIGSAWS. Therefore, the data augmentation technique applied on video data helps train the model on a diverse data.

5.2.2 Top-n Accuracy

We estimated the top-1, top-2, top-3 most probable gestures for both the tasks in MiroSurge dataset.

top-n Accuracy						
Task	Top-1		Top-2		Top-3	
	LOUO	LOTO	LOUO	LOTO	LOUO	LOTO
Band Twist- kin	35.15	39.23	56.04	59	70.77	73.35
Band Twist- kin+vid	68.052	76.51	80.21	86.61	85.45	90.94
Band Twist- vid	66.52	72.54	77.63	82.19	83.06	86.06
Weaving- kin	62.51	652	82.45	84.33	87.11	89.3
Weaving- kin+vid	73.95	75.12	85.81	87.23	89.57	91.2
Weaving- vid	73.2	73.2	84.64	84.99	87.96	89.18

Table 5.9: Top-n accuracy for MIRO dataset

5.2.3 Visualization

The figures below represent the timeline visualization for Weaving task performed in MIROSurge dataset. The top predictions for video, video and kinematic and only kinematic trained by leaving one user out is depicted below.

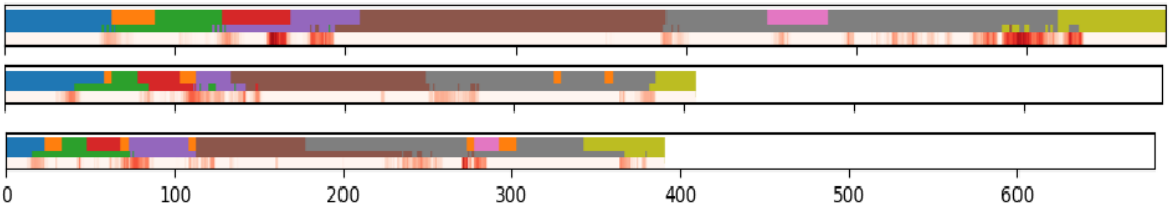

Figure 5.9: Weaving task prediction based on Video data

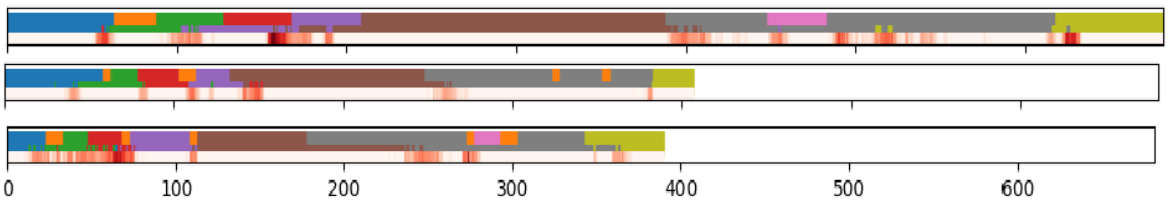

Figure 5.10: Weaving task prediction based on kinematic and video data

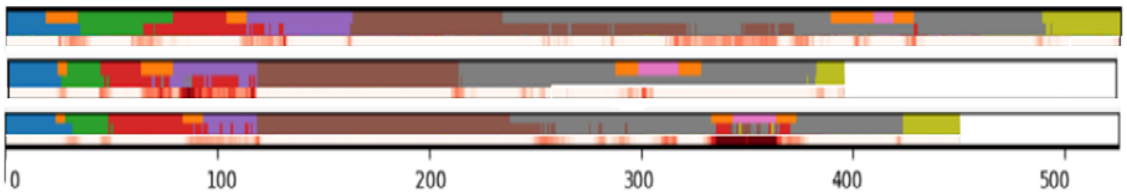

Figure 5.11: Weaving task prediction based on kinematic

All the above visualization depict the timeline for User 5's three trials. We perform MC Dropout to estimate the uncertainty. Since the amount of data is less, performing bootstrap ensemble as discussed in section 4.4.1 is not feasible.

6 Conclusion

Recognition of surgical gestures during minimally invasive surgery has to be performed accurately with high confidence. But the classification of such fine-grained actions cannot be performed using only contextual cues. Capturing nuances like the difference in the position of the needle, location of right and left arm, etc. is important for decision making. In this work, we have tried to explore models that can achieve this task with high confidence. Variations in the model's performance based on the type of data is also evaluated along with an estimate of uncertainty.

Surgical gestures can be considered as a sequence of actions performed to complete a task. Since each gesture performed has a relation with the gestures completed in the past and gestures to be performed; the first choice of a deep learning model to conduct experiments with was a Bidirectional LSTM. The model performs reasonably well on gestures it has seen. However, it fails along segment borders and in recognizing rare or unseen gestures. The current known state-of-the-art model, trained using video data is a variation of temporal convolution network called Temporal Deformable Residual Network. We also tried to reproduce the results of the Encoder-Decoder Temporal Convolutional Network. The slight variation in the accuracy achieved is due to the difference in methods of feature extraction applied initially on the image frames.

The Bidirectional LSTM model was further modified to be trained using both kinematic and video data.The resulting model outperformed the results of models trained using only video and kinematic data on the MIRO Surge dataset. On JIGSAWS, however, the MLP+Bidirectional LSTM model achieved an accuracy lower than the Bidirectional LSTM model trained on kinematic data. This could be because of the lesser number of kinematic features in the MIRO Surge dataset compared to JIGSAWS.

Apart from estimating the most probable gesture for a particular frame we also evaluated the accuracy for the top-3 most probable gestures. The top-3 most probable gestures provide the surgeons with options to choose from. It reduces the search space for the surgeons. As in, the surgeon needs to consider only the top-3 gestures amongst the 15 or more gesture labels available for a surgical task. This helps reduce the cognitive workload on the surgeons. The results also help prove that the model learns the gestures, as it's a part of top-3 gestures but at times the error in prediction could be due to the close similarity between the gestures.

To ascertain the model's learning capabilities, we applied 4 different confidence estimation techniques on the model. The MC Dropout, Naïve Ensemble, and Ensemble using Random Prior techniques estimated the epistemic uncertainty of the model

accurately, in most cases. But we got the best results with the Ensemble using Bootstrap. The ensemble provided a lucid estimate to aleatoric and epistemic uncertainty. The reduction in confidence of the model along the edges of gesture segments was useful to understand what the model doesn't know.

Gesture recognition has numerous applications in tasks like sign language recognition, virtual environment regulator, remote robot controller, etc. In this work, we have attempted to learn the most important aspects in terms of surgical robotics. We have tried to understand the type of data important to improve learning. Refined the user interface information by providing more information about probable gestures. Finally, an attempt to understand what the model knows.

6.1 Future Work

In this section, we discuss the possible future work to improve the task of recognition and segmentation of surgical gestures.

- There are numerous possible networks to process sequential data. To improve the prediction accuracy and edit distance for a model trained only on video data, a transformer network could be experimented with. A transformer network is generally used to convert an input sequence to an output sequence. It has many applications in machine translation tasks. Regular RNN or LSTM cannot handle long-range dependencies well and the processing is done sequentially; this is time-consuming. A transformer network can handle long term dependencies better and also has scope for parallelization. Handling long term dependencies is important because at times the surgeons may perform incorrect gestures. By knowing the long term contextual information, the model can help correct the surgeons' inadvertent errors.
- We require more sophisticated networks to process both kinematic and video data.
- Currently the feature extraction and model training is considered as separate processes. In the future, a model with end-to-end training can be used.
- The current method predicts the gestures in an offline scenario. The model should be adapted to process the data collected during minimally invasive surgery in real-time.
- MIROSurge dataset is a great initiative to improve the learning of a model trained for surgical gesture recognition. The dataset can be improved further by recording data for various other surgical tasks and improving the data quality and rate of data logging.

List of Figures

List of Tables

Bibliography

[1] L. Zhu and N. Laptev. "Deep and Confident Prediction for Time Series at Uber". In: *2017 IEEE International Conference on Data Mining Workshops (ICDMW)* (2017), pp. 103–110.

[2] S. Bodenstedt, D. Rivoir, A. Jenke, M. Wagner, S. T. Mees, J. Weitz, and S. Speidel. "Active learning using deep Bayesian networks for surgical workflow analysis". In: *International Journal of Computer Assisted Radiology and Surgery* 14 (2018), pp. 1079–1087.

[3] N. Ahmidi, L. Tao, S. Sefati, Y. Gao, C. Lea, B. B. Haro, L. Zappella, S. Khudanpur, R. Vidal, and G. D. Hager. "A Dataset and Benchmarks for Segmentation and Recognition of Gestures in Robotic Surgery". In: *IEEE Transactions on Biomedical Engineering* 64 (2017), pp. 2025–2041.

[4] M. Sensoy, M. Kandemir, and L. Kaplan. "Evidential Deep Learning to Quantify Classification Uncertainty". In: (June 2018).

[5] C. Lea, R. Vidal, and G. D. Hager. "Learning convolutional action primitives for fine-grained action recognition". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. 2016, pp. 1642–1649.

[6] C. Lea, A. Reiter, R. Vidal, and G. Hager. "Segmental Spatio-Temporal CNNs for Fine-grained Action Segmentation and Classification". In: (Feb. 2016).

[7] C. Lea, R. Vidal, A. Reiter, and G. D. Hager. "Temporal Convolutional Networks: A Unified Approach to Action Segmentation". In: *ECCV Workshops*. 2016.

[8] W.-H. Lee, J. Ortiz, B. Ko, and R. Lee. "Time Series Segmentation through Automatic Feature Learning". In: (Jan. 2018).

[9] L. Ding and C. Xu. "TricorNet: A Hybrid Temporal Convolutional and Recurrent Network for Video Action Segmentation". In: (May 2017).

[10] I. Funke, S. Mees, J. Weitz, and S. Speidel. "Video-based surgical skill assessment using 3D convolutional neural networks". In: (Mar. 2019).

[11] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. "Deformable Convolutional Networks". In: Oct. 2017, pp. 764–773. DOI: 10.1109/ICCV.2017.89.

[12] C. Feichtenhofer, A. Pinz, and R. Wildes. "Spatiotemporal Residual Networks for Video Action Recognition". In: (Nov. 2016).

[13] M. Althoff, C. Guernic, and B. Krogh. "Reachable set computation for uncertain time-varying linear systems". In: Jan. 2011, pp. 93–102. DOI: 10.1145/1967701.1967717.

[14] T. Team. *A Tragic Loss*. 2016.

[15] Z. Lipton. "A Critical Review of Recurrent Neural Networks for Sequence Learning". In: (May 2015).

[16] Colah. *Understanding LSTM Networks*. 2015.

[17] J. S. Sepp Hochreiter. *Long short-term memory*. Neural computation,MIT Press,Volume 9, 1997.

[18] Ö. yıldırım. "A novel wavelet sequence based on deep bidirectional LSTM network model for ECG signal classification". In: *Computers in Biology and Medicine* 96 (Mar. 2018). DOI: 10.1016/j.compbiomed.2018.03.016.

[19] Y. G. Alex Kendall. *What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?* 31st Conference on Neural Information Processing Systems (NIPS 2017), 2017.

[20] Y. Gal. *Uncertainty in Deep Learning*. 2016.

[21] C. E. Shannon. *A mathematical theory of communication.* Bell System Technical Journal, 27(3):379–423, 1948.

[22] R. M. Neal. *Bayesian learning for neural networks*. Springer Science and Business Media.(Vol. 118), 2012.

[23] Z. G. Yarin Gal. *Dropout as a Bayesian Approximation:Representing Model Uncertainty in Deep Learning*. JMLR: W and CP volume 48, 2016.

[24] A. C. Ian Osband John Aslanides. *Randomized Prior Functions for Deep Reinforcement Learning*. Neural Information Processing Systems NIPS 2018, 2018.

[25] G. et al. *The JHU-ISI gesture and skill assessment working set (JIGSAWS): A surgical activity dataset for human motion modeling.* Modeling and Monitoring of Computer Assisted Interventions (M2CAI) – MICCAI Workshop, 3, 2014.

[26] E. Karademir. *Motion Primitive Learning for Robot-Assisted Surgery*. 2018.

[27] G. Bradski. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000).

[28] D. et al. *Recognizing Surgical Activities with Recurrent Neural Networks*. 2016.

[29] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager. "Temporal Convolutional Networks for Action Segmentation and Detection". In: *CoRR* abs/1611.05267 (2016). arXiv: 1611.05267. URL: http://arxiv.org/abs/1611.05267.

[30] P. Lei and S. Todorovic. "Temporal Deformable Residual Networks for Action Segmentation in Videos". In: Apr. 2018. DOI: 10.1109/CVPR.2018.00705.

[31] B. Lakshminarayanan, A. Pritzel, and C. Blundell. *Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles*. 2016. arXiv: 1612.01474 [stat.ML].

[32] I. Osband, J. Aslanides, and A. Cassirer. "Randomized Prior Functions for Deep Reinforcement Learning". In: *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*. NIPS'18. Montréal, Canada: Curran Associates Inc., 2018, pp. 8626–8638. URL: http://dl.acm.org/citation.cfm?id=3327757.3327952.

[33] L. Tao, L. Zappella, G. D. Hager, and R. Vidal. "Surgical Gesture Segmentation and Recognition". In: *Medical image computing and computer-assisted intervention : MICCAI. International Conference on Medical Image Computing and Computer-Assisted Intervention* 16 Pt 3 (2013).